Convergence speed of a link-state protocol for IPv6 router autoconfiguration

Guillaume Chelius, Eric Fleury, Bruno Sericola, Laurent Toutain

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Convergence speed of a link-state protocol for IPv6 router autoconfiguration

Guillaume Chelius — Eric Fleury — Bruno Sericola — Laurent Toutain

N° 5701

Septembre 2005

Thème COM

ISSN 0249-6399 ISRN INRIA/RR--5701--FR+ENG

Convergence speed of a link-state protocol for IPv6 router autoconfiguration

Guillaume Chelius , Eric Fleury , Bruno Sericola , Laurent Toutain

Thème COM —Systèmes communicants
Projets Ares et Armor

Rapport de recherche n° 5701 —Septembre 2005 — 13 pages

Abstract: This report presents a model for the NAP protocol, dedicated to the auto-configuration of IPv6 routers. If the auto-configuration of hosts is defined by IPv6 and mandatory, IPv6 routers still have to be manually configured. In order to succeed in new networking domains, a full auto-configuration feature must be offered. NAP offers a fully distributed solution that uses a link state OSPFv3-like approach to perform prefix collision detection and avoidance. In this report, we present a model for NAP and analyze the average and maximum autoconfiguration delay as a function of the network size and the prefix space size.

Key-words: IPv6 router auto-configuration, convergence speed analysis, Markov models

Unité de recherche INRIA Rhône-Alpes

Temps de convergence d'un protocole à état de lien pour l'auto-configuration de routeurs IPv6

Résumé : Ce rapport présente une modélisation du protocole NAP prou l'auto-configuration des routeurs IPv6. SI l'auto-configuration des hôtes est standardisée dans IPv6, aucun mécanisme n'est cependantr proposé pour la configuration des routeurs IPv6. Afin de permettre un réel déployement du protocole notamment dans des domaines comme le *home networking*, un mécanisme complet d'auto-configuration du réseau doit être proposé. NAP propose une solution entièrement distribuée utilisant une approche à état de lien de type OSPFv3 afin d'éviter et de détecter les collisions entre les préfixes auto-configurés. Dans ce rapport, nous proposons une modélisation de NAP ainsi qu'une analyse du délai moyen et maximum de convergence du processus d'auto-configuration en fonction de la taille du réseau et du nombre de préfixes disponibles.

Mots-clés : auto-configuration des routeurs IPv6, analyse de temps de convergence, modèle markovien

1 Introduction

The standardization of IPv6 is almost over. New domains for networking and applications like Home Networking or Mobile Telephony are rising. These domains, generally targeting large audiences, require a large amount of addresses. Home networking is a very good example of an emerging application. Even if the size of the network is limited to a house or an apartment, the network topology may be complex due to several technologies used to transport informations (IEEE 1394 may be used to interconnect audio-video equipments, power lines may be present to control equipments like light bulb or shades, 802.11 wireless networks or the Bluetooth technology may be chosen to connect computers, printers and PDA,...). Cabling constraints may also be very restrictive in a home environment. The topology of the network may evolve dynamically, as for example, Bluetooth or 802.11 devices may leave or enter the network. The dynamism of the network and the lack of networking knowledge may lead to the creation of physical loops in the topology. The home network may also be multi-homed as several accesses (GPRS, ADSL, . . .) managed by different ISPs may be available at the same time. To sum up, with the development of information transport technologies and the introduction of IPv6 stacks in all devices, even home networks will evolve towards complex networks.

Auto-configuration has been presented as one of the attractive new features of IPv6. If these mechanisms simplify a lot the network management, they cannot be assimilated as real plug and play networking since routers still need to be manually configured. To succeed in new domains such as home networking, basic IPv6 auto-configuration processes have to be enhanced and new protocols have to be defined to complete the already standardized auto-configuration mechanisms. This article focuses on layer-3 auto-configuration (*i.e.* to provide every equipment connected in an home network - routers PC, . . . - with a global IPv6 address) and studies the stability of automatically allocated prefixes/addresses in an arbitrary network topology. We analytically study the performance of NAP[2], (*No Administration Protocol*), a distributed link-state protocol for IPv6 routers auto-configuration, and show that its convergence time remains short even in critical situations, *i.e.* when the number of routers to auto-configure is close to the number of available prefixes.

After this short introduction, section 2 focuses on IPv6 addresses structures, auto-configuration mechanisms and presents the issue of IPv6 routers auto-configuration. Section 3 describes our solution, the NAP protocol. Section 4 proposes a model for NAP as well as analytical results on the convergence speed of the protocol. Section 5 compares these results to simulations and finally, section 6 concludes the article.

2 Autoconfiguration functionalities

Auto-configuration is made possible by the large size of IPv6 addresses (128 bits). In this section, we shortly detail two apsects of the auto-configuration process, namely host auto-configuration and router auto-configuration.

2.1 IPv6 address and host auto-configuration

The main difference between IPv4 and IPv6 is the change in the addressing format. As for IPv4, the size of addresses remains fixed but is now 128 bits long. The growth of the IP addresses size leads to a potentially unlimited address space. However, in both versions of the IP protocol, the addressing scheme is hierarchical and allocation rules take into account administrative factors and routing efficiency. In consequence, all addresses may not be used and, in the case of IPv6, auto-configuration mechanisms are responsible for a waste of addresses.

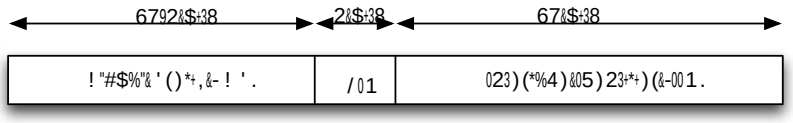

Figure 1: IPv6 adress format.

From a user point of view (Figure 1), an IPv6 unicast address may be seen as the concatenation of 3 parts [6]:

- the left most 48 bits: they contain the global routing prefix (GP). Historically called TLA (*Top Level Aggregator*), it is the prefix allocated to a network by the provider. The GP (*Global Prefix*) respects the administrative procedures for prefix delegation which are very similar to CIDR [5] in IPv4.
- the next 16 bits: they contain the *Subnet-ID* (SID) of the *Site* (called *Site Level Aggregator* (SLA) in previous documents). *Subnet-IDs* are used to number sub-networks inside a site. Values are chosen locally by the site manager. The addressing can be either flat or hierarchical. Our work focuses on automatic configuration of the SID inside a Home Network.
- the remaining 64 bits: they are dedicated to the *Interface IDentifier* (IID). This value is generally derived from the interface MAC address but can also be a random value to increase host privacy or bound to a public key in order to provide authentication. The huge size of this identifier reduces considerably the risk of collisions when several equipments auto-configure their addresses on a same link.

These lengths for the global routing prefix, the *Subnet-ID* and the *Interface ID* are not explicitly fixed by [6]. The size of the Subnet-ID is currently being discussed at the IETF (*Internet Engineering Task Force*) and in Registries forums. This value is well fitted for large organizations, but will not be efficient for Home Networking. A large number may ease subnetwork auto-numbering, like for the IID part, but it is unlikely that providers will continue to support this scheme. One can imagine that the size of the Subnet-ID field will be a commercial negotiation between a customer and its provider. However, its impact on auto-configuration protocols is important as the number of sub-networks in a site may become very close to the numbering capacity provided to the customer by the provider.

The *Neighbor Discovery Protocol* (NDP [9]) defines the way addresses are allocated to network interfaces. During the bootstrap period, a host creates a link-local address for each of its interfaces by concatenation of the well-known link-local prefix (`fe80::/16`) and the considered interface identifier. Then, the interface proceeds to a *Duplicate Address Detection* (DAD) to ensure that no other interface has the same address on the link. The link-local address allows the equipment to talk with other devices on the same local link and, in particular, with directly connected routers. If the routers are configured, they own a global prefix that they advertize. The host concatenates the global prefixes delivered by the link routers with its interface identifier in order to obtain global IPv6 addresses.

2.2 Router auto-configuration

Neighbor Discovery and DHCPv6 protocols are only the first step towards a full auto-configuration since they suppose that routers are already configured. A second step toward zero-networking is to automatically learn the GP assigned by ISPs to the network. Recent improvements to the DHCPv6 [10] protocol allow the ISP to inform the first router (CPE: *Customer Premise Equipment*) of the allocated global prefix.

The Subnet-ID value is more difficult to assign automatically since it reflects the site internal topology. A simple solution is possible if all interfaces are directly connected to a CPE router. This router acquires the global prefix from the ISP and automatically appends a Subnet-ID value based on the interface number. This model is very restrictive in term of topology and in terms of multi-homing capabilities as different ISPs will have to share the same CPE.

When the network topology is quite complex or evolves dynamically, L3 forwarding appears to be the best solution. In [2], we study the different ways Subnet-IDs may be dynamically allocated. More particularly, in this paper, we present and analytically study performances of the NAP (*No Administration Protocol*) proposal which uses a link-state approach to propagate prefix informations and establish a consensus on auto-configured unique Subnet-ID values.

3 No Administration Protocol for router self configuration

At this step of standardization, if IPv6 proposes auto-configuration protocols for hosts and end equipments, routers still need to be manually configured as the SID value of a link is not obtained automatically. To address this last point, we propose NAP, a protocol which, distributed among the routers of a site, allocates automatically a unique SID to each link. NAP is a distributed protocol that considers a link-state approach to perform IPv6 prefix auto-configuration. It was originally implemented as an extension to OSPFv3 [3] under the dslac [1] name. This protocol is relatively simple, based on reliable broadcast operations inside the network.

3.1 A distributed link-state protocol

NAP reuses OSPFv3 characteristics for distributed database management. The NAP LSA database is included in the OSPFv3 one using opaque types. On each auto-configured link, a *Designated*

Router (DR) is elected according to the OSPF election protocol and reliable database updates are performed according to the OSPF mechanism. The DR is responsible for choosing and assigning the IPv6 prefix to the link and for advertising this IPv6 prefix attribution to all auto-configured routers in the whole network, *i.e.*, for adding a new LSA entry in the NAP database containing the identity of the newly auto-configured link and the assigned prefix.

NAP characteristics derive from the OSPF ones. The main difference between NAP and OSPF is the *router id*s. It is a manually assigned 32 bits value in OSPFv3 while it is a 128 bits random value in NAP and/or one may suppose that each router is also set up with an unique ID during its manufacturing. The reason for this change is that we do not want NAP to request any manual configuration.

3.2 Description of the NAP algorithm

Let assume as stated above that each router is set up with an unique ID. The first stage of the protocol is to propagate inside the site, the site GP values learned by edge routers through the use of the DHCPv6 protocol. The second stage of the protocol, like for classical link-state routing protocols such as OSPF, is to elect a Designated Router on each link of the network. This router is responsible for selecting the SID value for the link and for announcing the chosen SID value to the other routers. As it knows the size and the value of the site GPs, each designated routers can find the size of the SID part needed to reach the 64 bits prefix boundary. Let m be the size in bit of the SID. Each DR draws a random number between 0 and $M = (2^m) - 1$ minus the already attributed values and then floods this value to all routers in the site. If no other router has selected the chosen value, the SID is allocated to the link. If several routers have selected the same value, the router with the lowest ID keeps it and the other routers have to renumber their link through the selection of another random value among those not already attributed. A more detailed description of the protocol can be found in [1] and [2].

Let L be the number of links to be numbered in a site and M the number of available SID values for the network. Of course we must have $L \leq M$. To each network link, we associate its LinkID value which is the concatenation of the link Designated Router ID and its link-connected interface number. A lexical order may be set among the links using their LinkID value. Let NCL (Non Configured Links) be the set of links with no valuable attributed prefix. This set is ordered using the lexical order defined by the LinkID values. The NAP auto-configuration process can be sequentially described using algorithm 1.

Algorithm 1 NAP

while $(NCL \neq \emptyset)$ **do**
  for $(i \in NCL)$ **do**
    $v(i)$ = random(SID);
  Broadcast;
  for $(i \in NCL$ such that $v(i) \in SID)$ **do**
    $NCL = NCL - \{i\}$;
    $SID = SID - \{i\}$;

4 Model and analysis

Given the very simple algorithm described above, we want to compute the stabilization speed of the NAP protocol in a network, that is, the mean number of steps that are required for each link to be attributed a unique SID value when the network is bootstrapped. The stabilization speed is thus directly correlated to the number of collisions that will occur and must be resolved, possibly in chain.

We model the NAP protocol by using a successive set of random drawns. The main goal is to assign a prefix to each link. A way to model this problem is to consider that the prefix interval is represented by a set of M urns in which one must randomly distribute L balls which are going to represent the links. As stated before the goal is to have only one ball (link) associated to a given urn (prefix). We thus have $L \leq M$.

The algorithm can be model in terms of urn/balls as follow:

Algorithm 2 NAP_PROCESS(M, L)

▷ *Input:* M *urns and* L *balls*
▷ *Pre condition:* $M \geq L$
if $(L \neq 0)$ **then**
  Randomly throw the L balls into the M urns;
1 Let $c \leq M$ denote the number of urns containing at least one ball;
2 Keep aside these c urns and for each of them one of the balls contained inside;
3 Call NAP_PROCESS($M - c$, $L - c$);
end

By repeating the procedure NAP_PROCESS, eventually every ball will be stored in an urn and every urn will contain at most one ball. Let us denote by N the random variable counting the number of iterations needed to reach such a configuration. N is the number of calls to the recursive procedure NAP_PROCESS including the first one. We are interested in computing the distribution and the expectation of the random variable N.

Let us consider the homogeneous discrete-time Markov chain $X = \{X_n,\, n \in \mathbb{N}\}$ on the state space $\mathcal{S} = \{0, 1, \ldots, L\}$ where the event $\{X_n = i\}$ represents, for $n \geq 1$, the fact that, after n transitions, or calls to the procedure NAP_PROCESS, the procedure NAP_PROCESS(L-i,M-i) is currently being executed. The Markov chain starts in state 0 with probability 1, which means that the first call to the procedure is made by NAP_PROCESS(L,M). We denote by $\mathbb{P}(L, M) = (p_{i,j}(L, M))_{(i,j)\in\mathcal{S}^2}$ the transition probability matrix of the Markov chain X. X is clearly acyclic and state L is the absorbing state of X. This means that for every $i \in \mathcal{S} - \{L\}$ and $j \in \mathcal{S}$, we have $p_{ij}(L, M) = 0$ for $i \geq j$ and $p_{L,L}(L, M) = 1$. The random variable N can thus be defined more formally, for $L \geq 1$, as

$$N = \sum_{n=0}^{L} \sum_{i=0}^{L-1} 1_{\{X_n = i\}}.$$

N is the number of transient states visited before absorption. Since the Markov chain is acyclic, we have $1 \leq N \leq L$, with probability 1.

In the next subsection, we derive the transition probability of matrix $\mathbb{P}(L,M)$ which has the following form, where we write $p_{i,j}$ instead of $p_{i,j}(L,M)$.

$$\mathbb{P}(L,M) = \begin{pmatrix} 0 & p_{0,1} & p_{0,2} & \cdots & \cdots & p_{0,L} \\ 0 & 0 & p_{1,1} & p_{1,2} & \cdots & p_{1,L} \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \cdots & 0 & p_{L-1,L} \\ 0 & 0 & \cdots & \cdots & 0 & 1 \end{pmatrix}.$$

4.1 Computation of the transition probability matrix

We are now going to derive the transition probability $p_{ij}(L,M)$ of the transition matrix $\mathbb{P}(L,M)$ for all $i,j \in \mathcal{S}$ such that $i<j$. For $i<j$, the transition probability $p_{i,j}(L,M)$ is the probability to obtain exactly $j-i$ non empty urns when throwing $L-i$ balls into $M-i$ urns.

For all $i<j$, we thus have:

$$p_{i,j}(L,M) \quad = \quad p_{0,j-i}(L-i,M-i). \tag{1}$$

We obtain these transition probablities using the following theorem.

Theorem 1 *The number of ways of throwing r different balls in n different urns such that exactly m urns are not empty is*

$$\binom{n}{m}\sum_{k=0}^{m}\binom{m}{k}(-1)^k(m-k)^r.$$

Proof. See for instance [4]. ■

The number of ways of throwing r different balls in n different urns being equal to n^r, we obtain, for $i<j$,

$$p_{i,j}(L,M) \quad = \quad \binom{M-i}{j-i}\sum_{k=0}^{j-i}\binom{j-i}{k}(-1)^k\left(\frac{j-i-k}{M-i}\right)^{L-i}. \tag{2}$$

As expected, we have $p_{L-1,L}(L,M)=1$ and the case $j=L$ yields to the well-known birthday problem and thus we have, for $i<L$,

$$p_{i,L}(L,M) = p_{0,L-i}(L-i,M-i) \quad = \quad \frac{(M-i)!}{(M-L)!(M-i)^{L-i}}. \tag{3}$$

4.2 Distribution of N

Let Q be the submatrix obtained from matrix $\mathbb{P}(L,M)$ by deleting the last line and the last column which correspond to the absorbing state L. We denote by α the row vector containing the initial probability distribution of the transient states of X. The dimension of vector α is thus equal to L and

since $X_0 = 0$ with probability 1, we have $\alpha = (1, 0, \ldots, 0)$. By using classical results on Markov chains, the distribution of N is given, for $n \geq 1$, by

$$\Pr\{N = n\} = \alpha Q^{n-1}(I - Q)\mathbb{1},$$

where I is the identity matrix of dimension L and $\mathbb{1}$ is the column vector of dimension L with all its entries equal to 1. The matrix Q being acyclic, we have $Q^L = 0$. The expected value of N is given by

$$\mathbb{E}(N) = \sum_{n=0}^{\infty} \Pr\{N > n\} = \sum_{n=0}^{\infty} \alpha Q^n \mathbb{1} = \alpha(I - Q)^{-1}\mathbb{1}.$$

To compute $\mathbb{E}(N)$ we first introduce the column vector $V = (V_i)_{0 \leq i \leq L-1}$ of conditional expectations $V_i = \mathbb{E}(N|X_0 = i)$, which is given by

$$V = (I - Q)^{-1}\mathbb{1}$$

Our goal is to compute $V_0 = \mathbb{E}(N)$. The vector V is solution to the linear system $(I - Q)V = \mathbb{1}$, which can also be written as $V = \mathbb{1} + QV$, that is, for every $0 \leq i \leq L - 1$,

$$V_i = 1 + \sum_{j=0}^{L-1} p_{i,j}(L, M)V_j.$$

The matrix $\mathbb{P}(L, M)$ being acyclic, we get, as expected, $V_{L-1} = 1$, and for every $0 \leq i \leq L - 2$,

$$V_i = 1 + \sum_{j=i+1}^{L-1} p_{i,j}(L, M)V_j.$$

The algorithm to compute $\mathbb{E}(N)$, which is equal to V_0, is the following :

Algorithm 3 MEAN NUMBER OF ITERATIONS(M, L)

▷ *Input: M urns and L balls*
▷ *Pre condition: $M \geq L \geq 1$*
$V_{L-1} = 1$;
for $i = L - 2$ **downto** 0 **do**
    $V_i = 1 + \sum_{j=i+1}^{L-1} p_{i,j}(L, M)V_j$;
end

The following table gives the average NAP convergence time for different network sizes and standard lengths of allocated GPs.

5 Simulations

We performed several simulations in order to validate the accuracy of the model we propose. We compared the analytical performance results to the simulation ones using an implementation `http://nap.dstm.info/`

GP length vs Network size	5 links	10 links	15 links	20 links	25 links
/60	1.5	2.19	3.06	*n.a.*	*n.a.*
/56	1.04	1.16	1.34	1.53	1.70
/52	1.00	1.01	1.03	1.05	1.07
/48	1.00	1.00	1.00	1.00	1.00

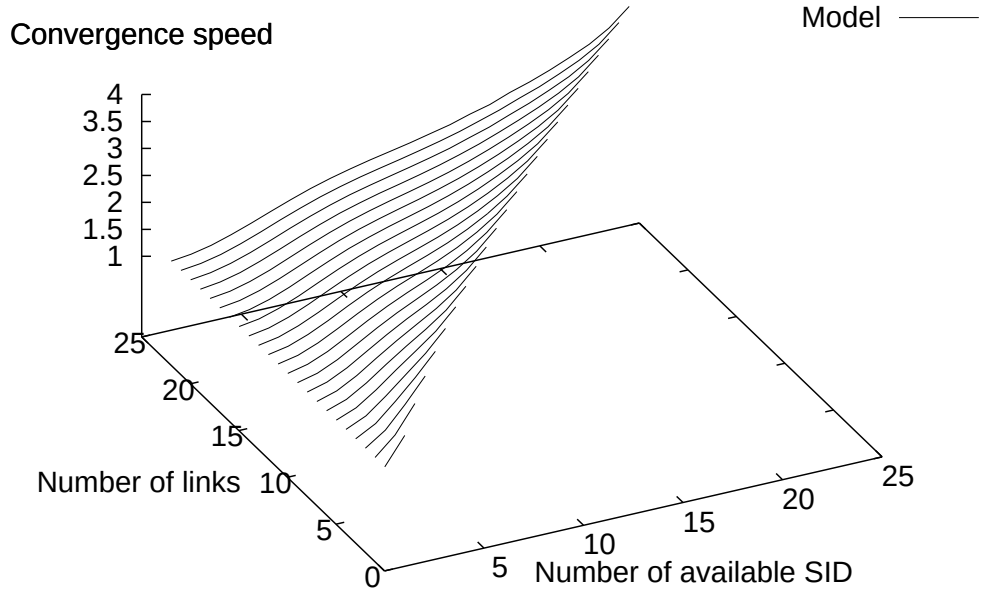

Figure 2: Analytic average convergence speed of the NAP protocol in number of rounds.

of the protocol. As in the previous section, the parameter we study is the convergence speed of the auto-configuration process. More precisely, it is the number of protocol rounds that are necessary to completely auto-configure the network. In term of network operations, for an unconfigured router, a round corresponds to the choice of a subnet value and its advertisement, *i.e.* a broadcast in the network. The convergence speed is a function of the considered network size and of the IPv6 prefix space (*i.e.*, the SubnetID space). Figure 2 shows that NAP achieves a really fast convergence. In average, no more than 4 rounds are needed to autoconfigure up to 25 routers even in a critical configuration, *i.e.* when the number of links to configure is equal to the number of available SubnetIDs.

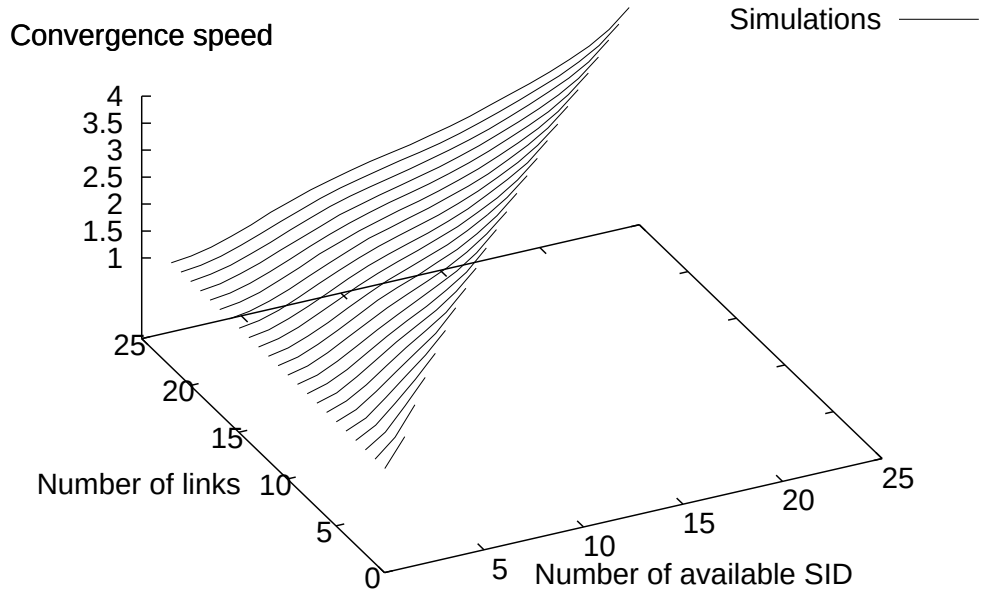

Figure 3: Average convergence speed of the NAP protocol in simulation (5000 iterations).

This fast convergence speed is confirmed by the simulation results presented in figure 3. They show indeed very similar performances. In order to simulate the NAP protocol, we have generated random network topologies using a free open source software designed for network topology analysis and modeling called network manipulator (`nem`) [8]. Nem is able to create realistic Internet-like topologies and can check proof them on the fly by a thorough topology analysis. We have generated topologies up to 25 routers. Finally, figure 4 presents the difference of results provided by the simulations and the analytical models. In the worst case, the difference is inferior to 0.015 rounds. As we can see, our presented model and results are validated by the simulations.

6 Conclusion and futur works

In this paper, we present a model for the NAP protocol. Based on this model, we compute the convergence speed of the protocol in the worst and average cases. The results are validated by simulations of the protocol. From these results, we can state that NAP presents a valid alternative to the centralized solutions like prefix delegation with DHCP [7]. Its convergence time remains short

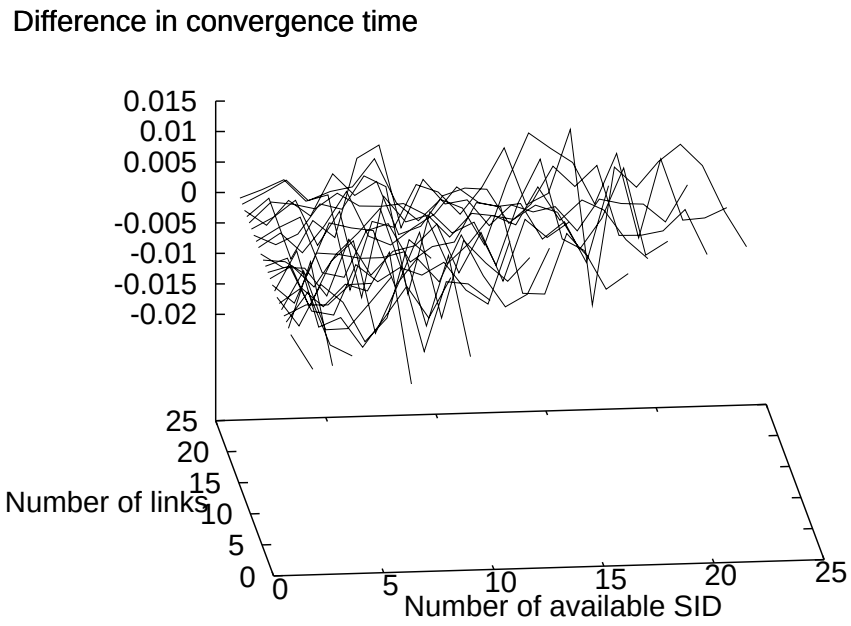

Figure 4: Difference between the simulation and the analytical results.

even in critical configurations, when the number of links is equal to the number of available prefixes. Moreover, its distributed architecture guarantees a high robustness and adaptability compared to centralized approaches. Moreover, as described in [2], it deals efficiently with multi-homing or network dynamic compared to the classical centralized approaches.

The study of NAP and the analysis that we developed in this article can be extended to other parameters such as the number of collisions that happen during the autoconfiguration process. From this result we can easily deduce the average number of flooding/broadcast operations required by NAP during the convergence process. This would give a good overview of the protocol impact on the network use. Other events than a network bootstrap can also be studied: the convergence time or the number of collisions after the merging of several autoconfigured networks for example.

References

[1] G. Chelius, É. Fleury, and L. Toutain. Using OSPFv3 for IPv6 for router autoconfiguration. Internet Draft draft-chelius-router-autoconf-00.tx, Internet Engineering Task Force, June 2002.

[2] G. Chelius, É Fleury, and L. Toutain. No admistration protocol (nap) for ipv6 router auto-configuration. *Int. J. Internet Protocol Technology*, 1(2):101–108, september 2005.

[3] R. Coltun, D. Ferguson, and J. Moy. OSPF for IPv6. IETF Request for Comments 2740, Internet Engineering Task Force, December 1999.

[4] M. Eisen. *Introduction to Mathematical Probability Theory*. Prentice Hall, Englewood Cliffs, 1969.

[5] V. Fuller, T. Li, J. Yu, and K. Varadhan. Classless inter-domain routing (cidr): an address assignment and aggregation strategy. RFC 1519, IETF, September 1993.

[6] R. Hinden, S. Deering, and E. Nordmark. IPv6 Global Unicast Address Format. IETF Request for Comments 3587, Internet Engineering Task Force, August 2003.

[7] T. Kniveton. Mobile network prefix delegation. Internet draft, IETF, July 2005.

[8] D. Magoni. nem: A Software for Network Topology Analysis and Modeling. In *10th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'02)*, pages 364–371, Fort Worth, Texas, USA, October 2002. IEEE/ACM.

[9] T. Narten, E. Nordmark, and W. Simpson. Neighbor Discovery for IP Version 6 (IPv6). IETF Request for Comments 2461, Internet Engineering Task Force, December 1998.

[10] O. Troan and R. Droms. IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6. IETF Request for Comments 3633, Internet Engineering Task Force, December 2003.

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifi que
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399